PhotoGrade – Fast and Effective Application for Digital Photo Editing

Hrvoje Ditrih, Sonja Grgić
University of Zagreb, Faculty of Electrical Engineering and Computing
Unska 3, 10000 Zagreb, Croatia
hrvoje.ditrih@fer.hr

***Abstract*—This paper presents a procedure for the development of an application that should be both a comprehensive photo editing tool and easy to use. The choice of programming language and libraries for digital image processing is explained. Principles of graphical user interface (GUI) design are presented. The tools inside the app are organized in such a way to enable user-intuitive means of photo editing and accelerate the workflow of experienced users. The features and capabilities of the application are presented through a detailed example of artistic photo editing alongside attached before and after of user-edited photos.**

***Keywords*—Digital Photography; Photo Editing; Image Processing; PhotoGrade; Artistic Photo**

I. Introduction

Photography is a form of documenting visual information, an irreplaceable way of recording important events, as well as a form of art expression. In addition to being used for technical, medical and business purposes, photography has an important role in recording everyday lives using smartphone cameras. Photographs are further processed with the purpose of picture quality improvements or for artistic reasons. Today, there are several software solutions for photo and image processing. Many applications provide such features. Some of the most popular are Photoshop [1] and Gimp [2], which provide comprehensive options for creating new image content or editing existing. Alongside them, some applications provide more specific functionalities that are targeted only at digital image editing. Such an application is Lightroom [3], which doesn't include features for creating digital images from scratch unlike Photoshop, but is focused on managing and adjusting images from digital cameras.

This paper aims to present a methodology for creating a simple and effective digital photo processing application that enables user processing of photos according to personal preferences quickly and easily. In line with that goal, an application named PhotoGrade (PG) was developed. The paper presents the complete process of application development from the selection of programming language and libraries for digital image processing, to graphical user interface design.

The application provides an intuitive way to edit photos with a range of digital image processing tools incorporated in GUI. To demonstrate the capabilities of PG, examples of edited images are shown. One detailed example is given with step-by-step instructions on how to edit the photo to achieve the artistic goal.

The paper is divided into four sections. After the introduction in the first section, the second section explains techniques, methods, and software solutions that have been used in the process of application development. The third section demonstrates the capabilities of applications when editing an image, either to improve image quality or achieve certain effects for artistic reasons. The fourth section gives conclusions of the paper and stresses the advantages and usefulness of the application for various purposes.

II. PhotoGrade Development

A. Languages and Frameworks

PG was developed in Java programming language [1]. The OpenCV [4] library extension for Java was used for image processing operations. GUI was developed with the Java Swing [5] framework which has native multi-platform support. Additionally, Math package from Apache Commons was used for cubic spline interpolation in some image processing methods and JFreeChart library was used for histogram visualization.

Java was chosen for its ability to run on any platform without the need for multiple recompilations, and its object-oriented paradigm which is naturally suited for GUI development. OpenCV was chosen for its wide range of image processing algorithms and its CUDA and OpenCL support.

B. Image Processing on Graphics Cards

When editing images, the user should see the image adjustments in real-time. For this purpose, it is recommended to process images on the graphics processing unit (GPU) rather than the central processing unit (CPU) because of its parallel processing architecture which can give a speed boost to image processing operations [2].

[1] https://www.photoshop.com/
[2] https://www.gimp.org/
[3] https://lightroom.adobe.com/

[4] https://opencv.org/
[5] https://docs.oracle.com/javase/8/docs/technotes/guides/swing/

C. PhotoGrade Functionalities

The primary purpose of the PG app is to enable the user to manipulate digital images whether it is to make slight technical adjustments or to completely and creatively reimagine the look of an image. The user can open an image of any resolution and most bit depths and formats. When finished with editing, one can use the exporting option to save an image to the permanent storage device in formats: png, bmp, gif, jpg, and jpeg.

Image editing is an iterative process of trial and error. For this purpose, well-established actions undo and redo are implemented to enable moving through the history of image adjustments. Furthermore, the user can compare the currently edited image with the original image at any point in editing. To provide more insight into the image, PG includes a histogram view (luminance and RGB) of the image which is refreshed automatically on image changes.

PG provides these tools for image editing: 2D geometric transformations, several tools for brightness and contrast management, gamma correction, color hue and saturation management, tinting, grayscale conversion, unsharp mask, gaussian blur, image inversion, and several predefined effects.

D. Image Processing Methods

This section will explain most image processing methods used by PG. Some well-known methods like Gaussian blur, gamma correction, and basic geometric transformations like rotation will not be explained. All methods assume 8-bit pixel intensities in RGB color space, that is, pixel intensities will be integer values in the range $[0,255]$. Most methods depend on parameters that are adjustable via the GUI.

1) Brightness and Contrast Management

Methods that control luminance are pixel transformations $g(i,j) = h(f(i,j))$, where i and j denote the row and the column of a pixel, $f(.)$ is an input image pixel, $g(.)$ is an output image pixel and $h(.)$ is a pixel transformation function [3].

a) Linear Method

A simple method to control brightness and contrast is an affine pixel transform function:

$$g(x) = af(x) + b, \tag{1}$$

where coefficient a is contrast and offset b is brightness [3].

b) Cubic Spline Method

This method is controlled by two parameters α and β which control contrast and brightness respectively. It is calculated by the expression

$$g(x) = g_\beta(g_\alpha(x)). \tag{2}$$

Functions $g_\alpha(x)$ and $g_\beta(x)$ are interpolated cubic spline curves between points (x_i, y_i). If parameter $\alpha > 0$ then $g_\alpha(x)$ is defined by five points: $(0,0), (64+\alpha, 64-\alpha), (127,127), (191-\alpha, 191+\alpha), (255,255)$. Otherwise, if $\alpha \leq 0$ then $g_\alpha(x)$ is defined by two points: $(0, 0-\alpha), (255, 255+\alpha)$. Function $g_\beta(x)$ is defined by three points: $(0,0), (127-\beta, 127+\beta), (255,255)$.

The cubic spline method for brightness and contrast management provides more natural results than the linear method. A plot of (2) for the case of contrast increase and brightness adjustment is shown in Fig. 1.

2) Shadows and Highlights Management

The idea of this method is to separately control brightness of darker and brighter parts of an image. Intensity range $[0, 85)$ represents shadows, range $[75, 170)$ represents midtones and $[170, 255]$ is highlights. To separately control brightness of the aforementioned ranges, cubic spline interpolation between seven points is performed: $(0,0), (42-a, 42+a), (85,85), (127-b, 127+b), (170,170), (212-c, 212+c), (255,255)$ where parameters a, b and c control brightness of shadows, midtones, and highlights respectively.

3) Levels

Levels are defined by parameters l and u. Intensities lower than l will transform into black pixels and intensities greater than u will transform into white pixels. Levels are defined by the expression:

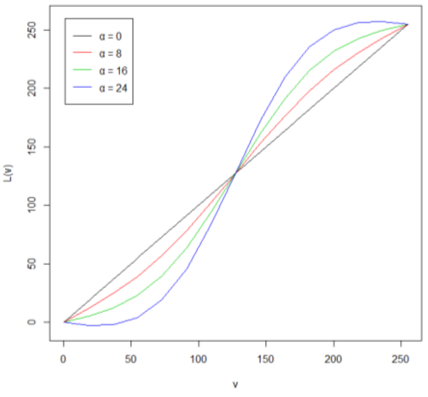
$$g(x) = \frac{255}{u-l} f(x) + \frac{255}{u-l} l. \tag{3}$$

4) Color Hue and Saturation Management

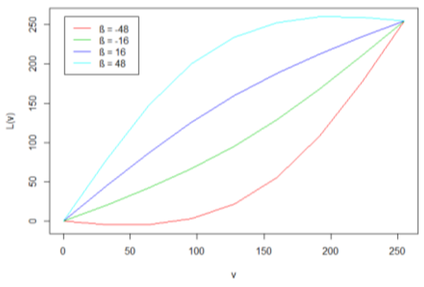
This method firstly transforms pixels from RGB to HSV color space and performs next pixel transformation:

$$g_{HSV}(x) = f_{HSV}(x) + b = (h + \Delta h, s + \Delta s, v + \Delta v), \tag{4}$$

where $f_{HSV}(x) = (h, s, v)$ is HSV triplet of input pixel and parameter $b = (\Delta h, \Delta s, \Delta v)$ controls hue, saturation, and value offset in HSV color space.



a) Contrast increase, $\beta = 0$



b) Brightness adjustment, $\alpha = 0$

Figure 1. Contrast and brightness management with cubic spline method – pixel transformation function plot

5) Tint

Tinting is a process of coloring an image with a single color. It is done in HSV color space by using the expression:

$$g_{HSV}(x) = (h_t, s_t, v(x)), \qquad (5)$$

where parameters h_t and s_t are hue and saturation of a tint color in HSV color space and $v(x)$ is an input pixel value of an HSV color space.

E. *Sepia Toning*

Sepia toning is a process of coloring a photograph with a brownish color to give it an old-fashioned or antiquated look. Considering every pixel of an image as a vector $[B \quad G \quad R]^T$ where B, G and R represent blue, green, and red component of an RGB color space, sepia toning is performed as a linear transformation of every pixel with a linear operator [4]:

$$\boldsymbol{A}_{sepia} = \begin{bmatrix} 0.272 & 0.534 & 0.131 \\ 0.349 & 0.686 & 0.168 \\ 0.393 & 0.769 & 0.189 \end{bmatrix}.$$

F. *Graphical User Interface*

PG can be run on any computer that has Java installed. To manipulate the images, the user works with the tools provided in PG. All tools operate identically so that the user can quickly learn how to use the app. The first step of editing is choosing a tool via the main menu (Fig. 2). Then, for most tools, an additional window pops up with adjustable sliders that control the tool parameters (Fig. 3). When adjusting the aforementioned parameters, changes to the image should be perceived instantly. Instead of accessing the tools via the main menu, every tool and action has assigned keyboard shortcut to provide the advanced users with a faster and more productive workflow. To enable working with images of all sizes, large images rescale automatically to fit the PG window.

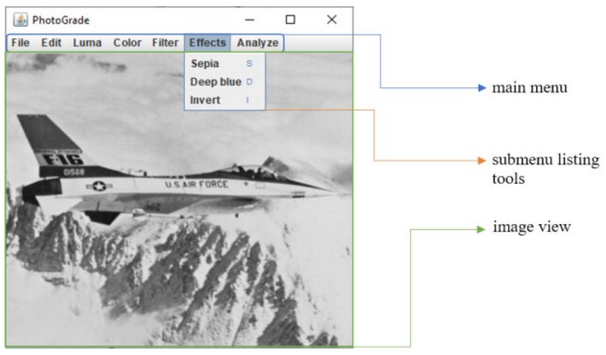


Figure 2. PhotoGrade main window

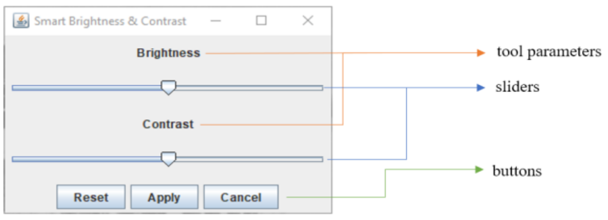


Figure 3. Parameters window

III. Image Editing with PhotoGrade

Common examples of image editing with PG are exposure correction, cropping, rotating, vertical or horizontal flipping, sharpening, and converting images to grayscale. 2D geometric transformation alongside brightness and contrast management tools are usually used for technical adjustments while combining them with other tools inside PG could result in a more drastic image transformation that is usually considered a form of an artistic photography expression.

A. *Image Editing Example*

This section will provide a walkthrough of steps in editing an image (Fig. 4a) for artistic reasons. Firstly, the image is converted to grayscale (Fig. 4b). Then, vertical flipping is applied to give an impression that the bug is walking (Fig. 4c). To make it look like the bug is glowing, an image inversion is applied (Fig. 4d). With brightness and contrast management tools the brightness is lowered to make the surroundings dark, and contrast is increased to highlight the details on the bug and make the glowing seem stronger (Fig. 4f). To change the color of the glow and give the image a more vibrant look, a yellow tint is applied which is further adjusted with color hue and saturation management tool (Fig. 4g).

B. *Images Edited by Users*

This section will show before and after of images edited by the author Hrvoje Ditrih and other users who tested PG. Each person mentioned in this section gave their consent to publish their names and edited images, to which they own the copyright. Names of authors, as well as their original and edited images, are shown in Fig. 5.

The author, Hrvoje Ditrih, demonstrates the creative use of PG (Fig. 5, a1-b2). Nevio Aradski (Fig. 5, c1-d2), Tihana Najdert (Fig. 5, f1-f2), and Emanuela Răileanu (Fig. 5, g1-g2) exemplify a more professional photo editing approach. Leona Turković (Fig. 5, e1-e2) and Vladimir Šeba (Fig. 5, h1-h3) play with color and body shapes.

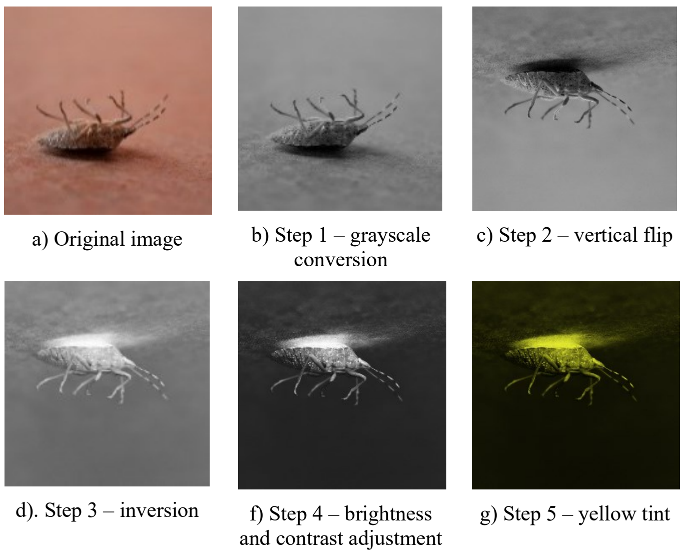
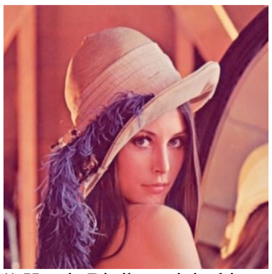
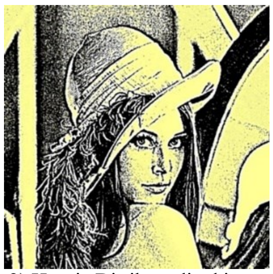
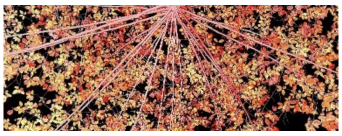

a) Original image
b) Step 1 – grayscale conversion
c) Step 2 – vertical flip
d). Step 3 – inversion
f) Step 4 – brightness and contrast adjustment
g) Step 5 – yellow tint

Figure 4. Image editing steps

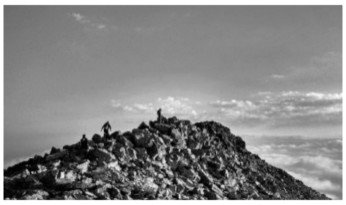

a1) Hrvoje Ditrih – original image


a2) Hrvoje Ditrih – edited image

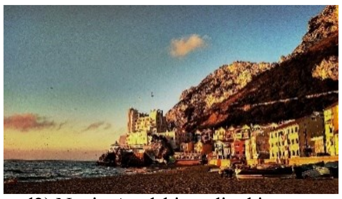
b1) Hrvoje Ditrih – original image


b2) Hrvoje Ditrih – edited image

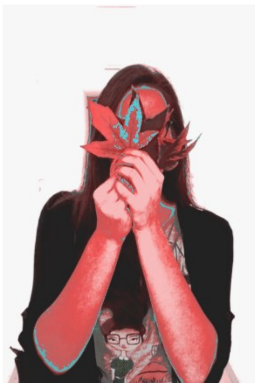
c1) Nevio Aradski – original image


c2) Nevio Aradski – edited image

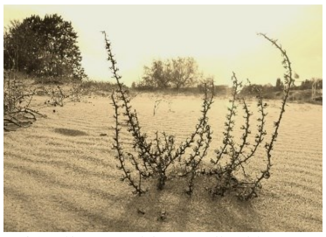
d1) Nevio Aradski – original image


d2) Nevio Aradski – edited image

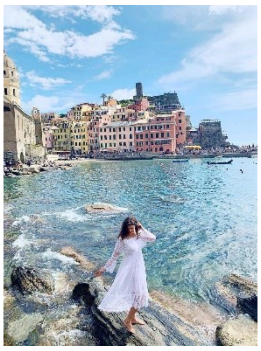
e1) Leona Turković – original image

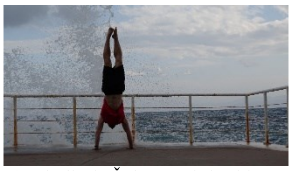

e2) Leona Turković – edited image

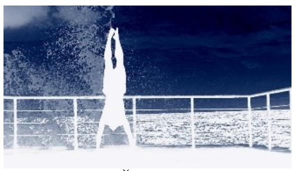
f1) Tihana Najdert – original image


f2) Tihana Najdert – edited image

g1) Emanuela Răileanu – original


g2) Emanuela Răileanu – edited image


h1) Vladimir Šeba – original image


h2) Vladimir Šeba – edited image

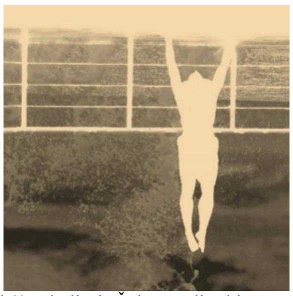

h3) Vladimir Šeba – edited image, version 2

Figure 5. Before and after of user-edited images in PhotoGrade

IV. Conclusion

PhotoGrade provides users with many basic functionalities for photo editing and enables them to complete many different requests. It is adequate for medium complex corrections of photos and excellent in the manipulation of brightness and contrast. Furthermore, it revives the user's imagination when using editing tools. Implemented tools for color adjustment, focusing, adding effects, geometric transformations and hue, color map, and brightness changes, allow the user to express their creativity. The tools are convenient for creating imaginative, unreal, abstract, and stylized images in which colors and shapes do not behave as in reality.

Future work on PhotoGrade should include adding more tools for photo manipulation and adding more application features. It should also include reworking and improving existing tools. Additionally, rewriting PhotoGrade in some compiled language like C++ is an option to compare the performance over an existing Java implementation.

References

[1] J. Bloch, Effective Java. Boston: Addison-Wesley Professional, 2018.

[2] Z. Yang, Y. Zhu and Y. Pu, "Parallel Image Processing Based on CUDA," 2008 International Conference on Computer Science and Software Engineering, Hubei, 2008, pp. 198-201, doi: 10.1109/CSSE.2008.1448.

[3] R. Szeliski, Computer Vision: Algorithms and Applications. London: Springer, 2010.

[4] A. Ahmadi, "How to Apply Sepia Filter to Images Using OpenCV," March, 2016. [Online]. Available: https://amin-ahmadi.com/2016/03/24/sepia-filter-opencv/. [Accessed May 8, 2020]